Question 1

I.Problem-solving approach

First, load the iris dataset with load_iris(), then the data is converted into DataFrame types and preprocess by using the pandas library, then the dataset is divided into training and test sets, and then build the model and use predict() to predict, and finally using recall_score(), precision_score(), and f1_score() to calculate the recall, accuracy and F1 score.

II.Libraries

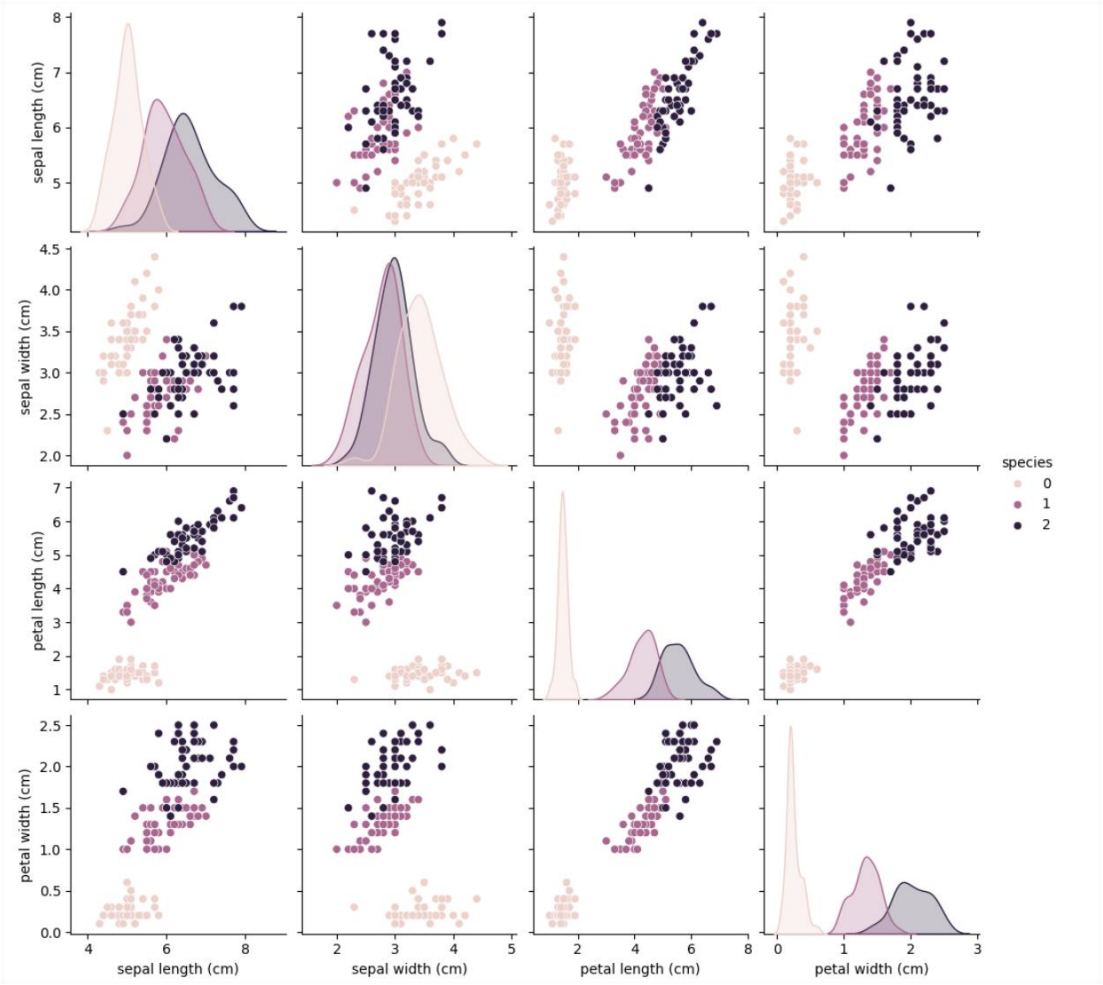
Use pandas to process data

Use seaborn to draw chart

Use sklearn to plot the decision binary tree and calculate recall, precision, and F1 scores

III.The result of the output

1.Pair Plot



2.binary decision trees(export_text())

```
|--- petal width (cm) <= 0.80
|   |--- class: 0
|--- petal width (cm) >  0.80
|   |--- class: 1
```

```
|--- petal width (cm) <= 0.80
|   |--- class: 0
|--- petal width (cm) >  0.80
|   |--- petal width (cm) <= 1.75
|   |   |--- class: 1
|   |--- petal width (cm) >  1.75
|   |   |--- class: 2
```

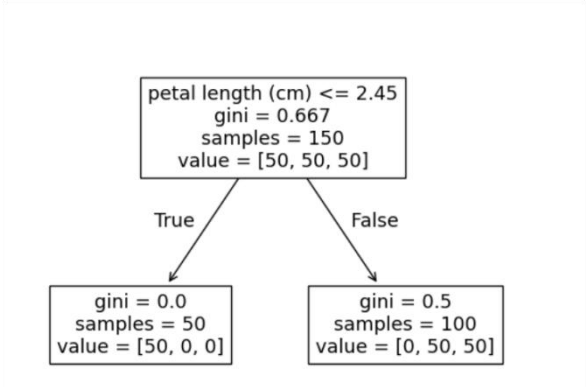
```
|--- petal length (cm) <= 2.45
|   |--- class: 0
|--- petal length (cm) >  2.45
|   |--- petal width (cm) <= 1.75
|   |   |--- petal length (cm) <= 4.95
|   |   |   |--- class: 1
|   |   |--- petal length (cm) >  4.95
|   |   |   |--- class: 2
|   |--- petal width (cm) >  1.75
|   |   |--- petal length (cm) <= 4.85
|   |   |   |--- class: 2
|   |   |--- petal length (cm) >  4.85
|   |   |   |--- class: 2
```

```
|--- petal width (cm) <= 0.80
|   |--- class: 0
|--- petal width (cm) >  0.80
|   |--- petal width (cm) <= 1.75
|   |   |--- petal length (cm) <= 4.95
|   |   |   |--- sepal length (cm) <= 4.95
|   |   |   |   |--- class: 1
|   |   |   |--- sepal length (cm) >  4.95
|   |   |   |   |--- class: 1
|   |   |--- petal length (cm) >  4.95
|   |   |   |--- petal width (cm) <= 1.55
|   |   |   |   |--- class: 2
|   |   |   |--- petal width (cm) >  1.55
|   |   |   |   |--- class: 1
|   |--- petal width (cm) >  1.75
|   |   |--- petal length (cm) <= 4.85
|   |   |   |--- class: 2
|   |   |--- petal length (cm) >  4.85
|   |   |   |--- class: 2
```

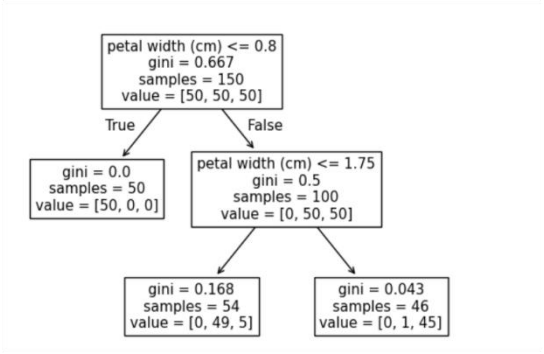
```
|--- petal length (cm) <= 2.45
|   |--- class: 0
|--- petal length (cm) >  2.45
|   |--- petal width (cm) <= 1.75
|   |   |--- petal length (cm) <= 4.95
|   |   |   |--- sepal length (cm) <= 4.95
|   |   |   |   |--- class: 1
|   |   |   |--- sepal length (cm) >  4.95
|   |   |   |   |--- class: 1
|   |   |--- petal length (cm) >  4.95
|   |   |   |--- petal width (cm) <= 1.55
|   |   |   |   |--- class: 2
|   |   |   |--- petal width (cm) >  1.55
|   |   |   |   |--- class: 1
|   |--- petal width (cm) >  1.75
|   |   |--- petal length (cm) <= 4.85
|   |   |   |--- class: 2
|   |   |--- petal length (cm) >  4.85
|   |   |   |--- class: 2
```

3.binary decision trees(plot_tree())

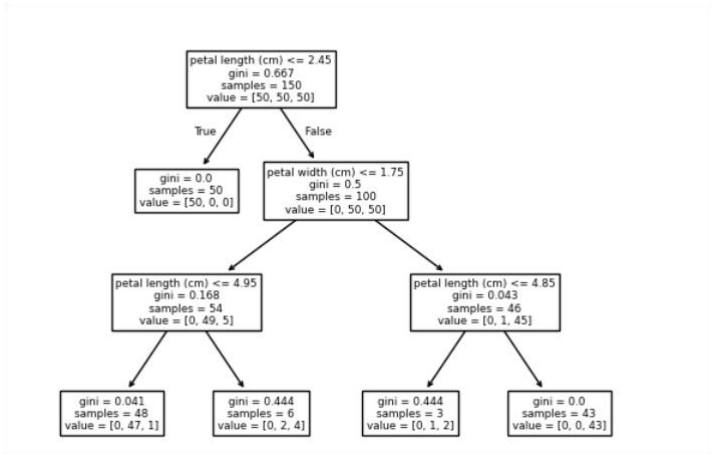
(1)max_depth=1



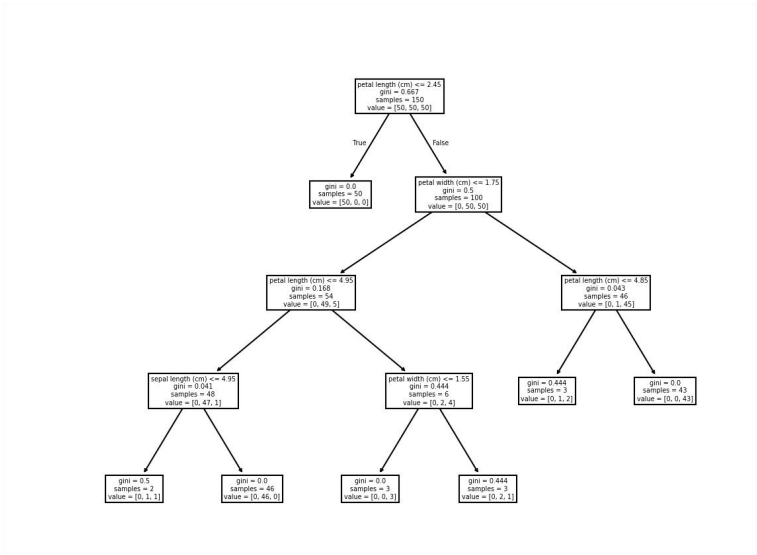
(2)max_depth=2



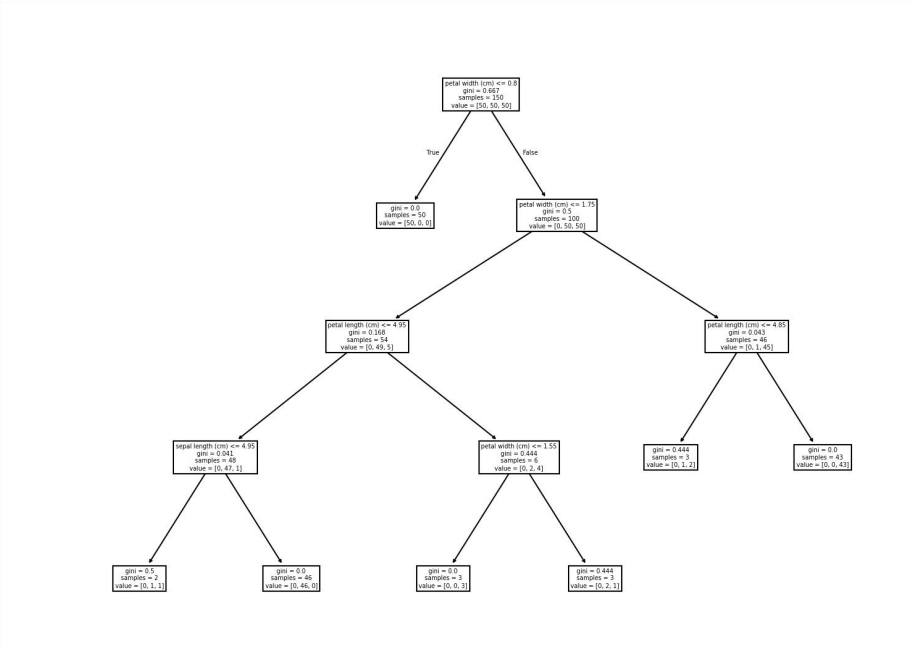
(3)max_depth=3



(4)max_depth=4



(5)max_depth=5



4.Calculation results

	max_depth	recall	precision	f1
0	1	0.711111	0.855556	0.614815
1	2	0.977778	0.979365	0.977745
2	3	1.000000	1.000000	1.000000
3	4	1.000000	1.000000	1.000000
4	5	1.000000	1.000000	1.000000

IV.Analysis of conclusions

After using the code in the jupyter notebook, we can get the following data：

	max_depth	recall	precision	f1
0	1	0.711111	0.855556	0.614815
1	2	0.977778	0.979365	0.977745
2	3	1.000000	1.000000	1.000000
3	4	1.000000	1.000000	1.000000
4	5	1.000000	1.000000	1.000000

As can be seen in the figure, when max_depth=5, there is the highest regression rate, because the deeper the tree depth is, the more complex the model will become, and the more details can

be captured, which improving the recall. The lowest accuracy is at max_depth=1, probably because the depth is too low so the detail which can be caught by the model is too little. When max_depth = 3, 4, 5 the F1 score is the best.

Micro-average：
Method: Ignore the differences for all categories, and directly add each category separately and then calculate the overall data.
Features: Treat each sample fairly, and be more affected by the category with a large sample size
Applicable case: When the category with a large sample size is relatively important

Macro-average：
Methods: Precision, recall, and F1 scores were calculated separately for each category, and then the values were taken as an arithmetic mean
Characteristics: Affected by categories with a small sample size
Use: Cases where categories with a small sample size are relatively important

Weighted-average：

Methods: The weighted average was weighted to average precision, recall, and F1 scores based on the sample size for each category.
Features: Treat each sample fairly, and be more affected by the category with a large sample size
Availability: When there is an imbalance in the category

Question 2

I.Problem-solving approach

First, load the Breast Cancer Wisconsin (Diagnostic) sample dataset through ucimlrepo, then use ucimlrepo to converted the data into DataFrame types , then the dataset is divided into training and test sets, and then build the model. After that, get the information about the first split. Then, Construct functions for calculating entropy, gini, and misclassification errors. At last, calculate entropy, Gini, and misclassification errors.

II.Libraries

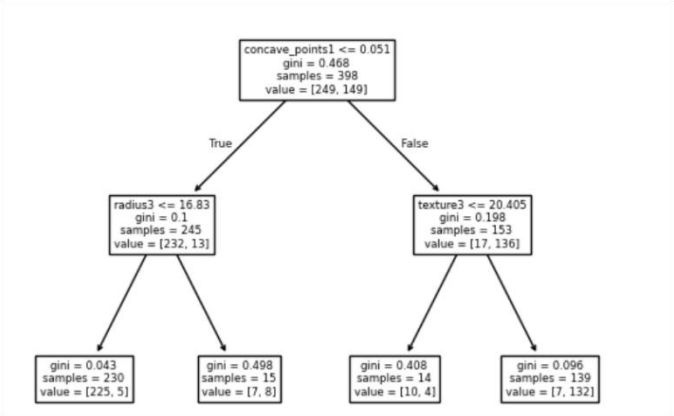
Use pandas to process data
Use numpy to calculate entropy, Gini, and misclassification errors
Use sklearn to plot the decision binary tree and create the model.
Use ucimlrepo to get the dataset.

III.The result of the output

1.binary decision trees(export_text())

```
Decision Tree Structure:
|--- concave_points1 <= 0.05
|   |--- radius3 <= 16.83
|   |   |--- class: 0
|   |--- radius3 >  16.83
|   |   |--- class: 1
|--- concave_points1 >  0.05
|   |--- texture3 <= 20.40
|   |   |--- class: 0
|   |--- texture3 >  20.40
|   |   |--- class: 1
```

2.binary decision trees(plot_tree())



3.The result of the output

```
First split feature: concave_points1
First split threshold: 0.05127999931573868

Information Gain: 0.5763
Gini: 0.4684
Misclassification Error: 0.3744
```

```
Parent Node Entropy: 0.9540
Left Child Node Entropy: 0.2993
Right Child Node Entropy: 0.5033
```

IV.Analysis of conclusions

By running the code in the jupyter notebook we can get:

```
First split feature: concave_points1
First split threshold: 0.05127999931573868

Information Gain: 0.5763
Gini: 0.4684
Misclassification Error: 0.3744


Parent Node Entropy: 0.9540
Left Child Node Entropy: 0.2993
Right Child Node Entropy: 0.5033
```

Through the result we can know that the entropy at the first split is reduced by 0.1514, indicating that the purity of the information is improved. The Gini of the first split is 0.4684, the information error is 0.3744, and the information gain is 0.5763. Select concave_pionts1 for the first split and determine the value of the decision boundary.

Question 3

I.Problem-solving approach

First, load the Breast Cancer Wisconsin (Diagnostic) sample dataset through ucimlrepo, then use ucimlrepo to converted the data into DataFrame types, then divide the dataset into training and test sets, and then perform PCA dimensionality reduction beforehand. After that build the model and use predict() to predict, At last use recall_score(), precision_score(), and f1_score() to calculate the recall, accuracy and F1 score, and calculate the values for False Positives (FP) and True Positives (TP), as well as the False Positive Rate (FPR) and True Positive Rate (TPR) through known formulas.

II.Libraries

Use pandas to process data
Use numpy to calculate values we want
Use sklearn to perform PCA dimensionality reduction and create the model.
Use ucimlrepo to get the dataset.

III.The result of the output

```
Using the first  principal component:
F1 Score: 0.9244
Precision: 0.9821
Recall: 0.8730
False Positives (FP): 1
True Positives (TP): 55
False Positive Rate (FPR): 0.0093
True Positive Rate (TPR): 0.8730
```

```
Using first 2 principal components:
F1 Score: 0.9244
Precision: 0.9821
Recall: 0.8730
False Positives (FP): 1
True Positives (TP): 55
False Positive Rate (FPR): 0.0093
True Positive Rate (TPR): 0.8730
```

IV.Analysis of conclusions

After running the code in the jupyter notebook, we can get the results as shown in the previous part, in which the F1 score is 0.9244, the accuracy is 0.9821, and the regression rate is 0.873 after PCA dimensionality reduction using only the first principal component. It shows that the model after PCA dimensionality reduction can effectively reflect the data content. After adding the second principal component, we can see that the F1 score, accuracy, and regression rate have not improved, so the first principal component already contains most of the information. With the two principal components, the FP was 1, the TP was 5, the FPR was 0.0093, and the TPR was 0.8730. The use of continuous data is beneficial because it captures more information and has a positive impact on model performance.