Lab 3

Decision Tree

1 Description

In this assignment, you are going to build a decision tree on the **UCI Breast Cancer Wisconsin (Diagnostic)** dataset, using the `scikit-learn` library.

The UCI Breast Cancer Wisconsin (Diagnostic) dataset is used for classifying tumors as malignant or benign based on 30 numerical features derived from imaging data. It includes 569 samples, with labels indicating either malignant (M) or benign (B).

Visit: `https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic`

2 Specifications

You are required to write a **Python Notebook** (.ipynb) and use `scikit-learn` library to complete the following tasks.

Although there are no strict rules for organizing the code, each task should be clearly documented and must fully satisfy all specifications.

2.1 Preparing the datasets

This task sets up the training and test datasets for the upcoming experiments.

You can download the UCI Breast Cancer Wisconsin (Diagnostic) dataset via Python as follow:

```
# Install the ucimlrepo package
!pip install ucimlrepo

# Import the dataset into your code
from ucimlrepo import fetch_ucirepo
breast_cancer_wisconsin_diagnostic = fetch_ucirepo(id=17)

# Data (as pandas dataframes)
feature = breast_cancer_wisconsin_diagnostic.data.features
label = breast_cancer_wisconsin_diagnostic.data.targets
```

With features and labels above, please prepare the following four subsets:

- `feature_train`: a set of training samples.
- `label_train`: a set of labels corresponding to the samples in `feature_train`.
- `feature_test`: a set of test samples, it is of similar structure to `feature_train`.
- `label_test`: a set of labels corresponding to the samples in `feature_test`.

You need to shuffle the data before splitting and split it in a stratified fashion. Other parameters (if there are any) should be left at their default settings.
There will be experiments on training and test sets with different proportions, including 40/60, 60/40, 80/20, and 90/10 (train/test); therefore, you will need 16 subsets.
Visualize the class distributions in all datasets (the original set, training set, and test set) across all proportions to demonstrate that they have been prepared appropriately.

2.2 Building the decision tree classifiers

This task involves conducting experiments on the designated train/test proportions listed above. You need to fit an instance of `sklearn.tree.DecisionTreeClassifier` (using information gain) to each training set and visualize the resulting decision tree with Graphviz.

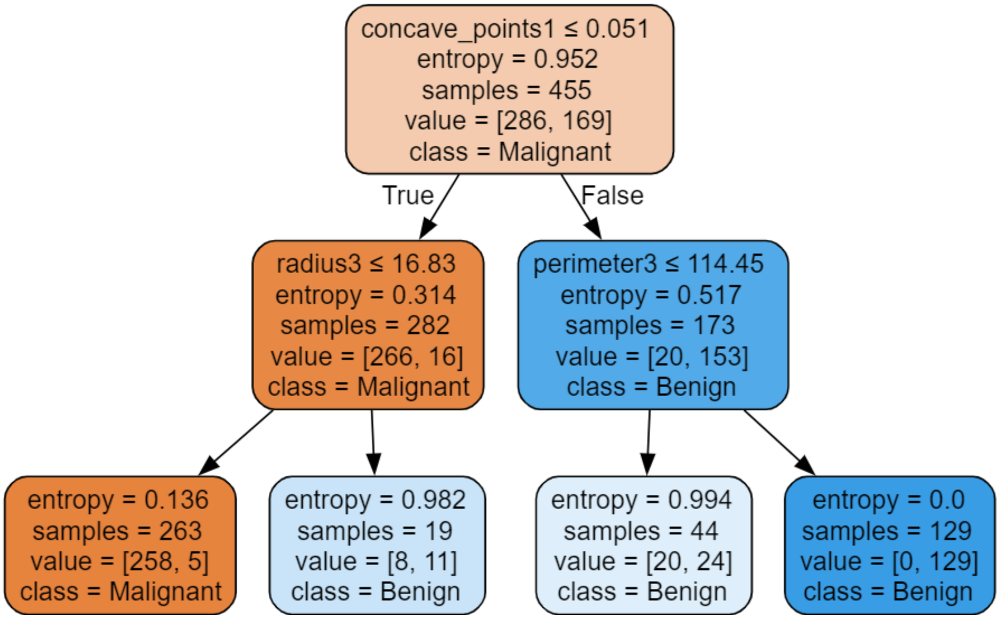
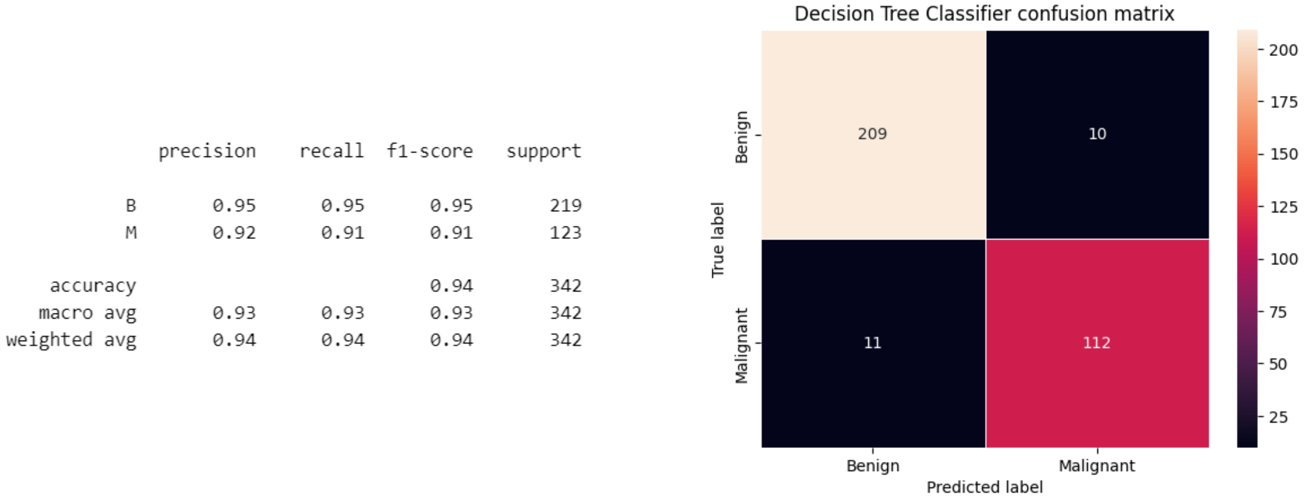


Figure 1: Example for a decision tree classifier (with depth = 2).

2.3 Evaluating the decision tree classifiers

For each of the above decision tree classifiers, predict the samples in the corresponding test set and generate a report using `classification_report` and `confusion_matrix`.



```
              precision    recall  f1-score   support

           B       0.95      0.95      0.95       219
           M       0.92      0.91      0.91       123

    accuracy                           0.94       342
   macro avg       0.93      0.93      0.93       342
weighted avg       0.94      0.94      0.94       342
```

Figure 2: Example for Classification Report and Confusion Matrix.

How do you interpret the classification report and the confusion matrix? Based on that, provide your comments on the performance of these decision tree classifiers.

2.4 The depth and accuracy of a decision tree

This task focuses on the 80/20 training and test sets. You need to consider how the depth of the decision tree affects classification accuracy.

You can specify the maximum depth of a decision tree by adjusting the `max_depth` parameter. You need to try the following values for parameter `max_depth`: None, 2, 3, 4, 5, 6, 7. And then:

- Provide the decision trees, drawn by Graphviz, for each `max_depth` value.
- Report the `accuracy_score` (on the test set) of the decision tree classifier for each value of the `max_depth` parameter in the following table.

`max_depth`	None	2	3	4	5	6	7
Accuracy							

- Provide comments on the statistics reported above.

3 Requirements

3.1 Report

The report must fully give the following sections:

- Your information (Student ID, full name, etc.).
- Self-evaluation of the completion rate of the lab and other requirements.
- All visualizations must be presented in the .ipynb file, while statistical results and comments are presented in the report.
- The report needs to be well-formatted and exported to PDF. Note that for editors like Jupyter notebook, you need to find a way to format it well before exporting to PDF.
- If there are figures cut off by the page break, etc., points will be deducted.
- References (if any).

3.2 Submission

- All reports, code, etc., must be contributed in the form of a compressed file (.zip, .rar, .7z) and named according to the format **StudentID.zip/.rar/.7z**.
- If the compressed file is larger than 25MB, prioritize compressing the report and source code. Images, etc., may be uploaded to the Google Drive and shared via a link.

4 Assessment

No.	**Details**	**Score**
1	Preparing the datasets	30%
2	Building the decision tree classifiers	20%
3	Evaluating the decision tree classifiers	
	Classification report and confusion matrix	10%
	Comments	10%
4	The depth and accuracy of a decision tree	
	Trees, tables, and charts	20%
	Comments	10%
	Total	**100%**

5 Notices

Please pay attention to the following notices:

- This is a **INDIVIDUAL** assignment.
- Duration: about 2 weeks.
- Any plagiarism, any tricks, or any lie will have a 0 point for the course grade.

The end.